B3.3-R4: SOFTWARE ENGINEERING & CASE TOOLS

NOTE:

1. **Answer question 1 and any FOUR from questions 2 to 7.**
2. **Parts of the same question should be answered together and in the same sequence.**

Time: 3 Hours **Total Marks: 100**

1.

a) Explain V-Model for software development with the help of a diagram.
b) What do you understand by component based software engineering?
c) Explain the significance of CASE tools.
d) Explain the difference between static testing and dynamic testing.
e) What is User Interface Design? Explain in brief?
f) What do you understand by configuration management?
g) Why software maintenance is important?

(7x4)

2.

a) What are the principles of clean room software engineering?
b) Why do we need software reusability? How do you ensure such reusability?
c) What do you mean by software agent? Explain the basic concepts of software agent?

(6+6+6)

3.

a) What do you understand by data dictionary? Explain with an example.
b) State the general organization of software requirement specification (SRS) for a large software project.
c) Explain Diagramming conventions of an ER diagram.

(6+6+6)

4.

a) What is a legacy system? Why is it necessary to re-engineer a legacy system? Explain using a schematic diagram, the main steps that you would undertake to re-engineer a legacy system.
b) How does software CBD resemble the use of components?
c) What are the considerations and importance of good software design?

(6+6+6)

5.

a) Explain the difference between software agent and program?
b) What do you understand by cohesion & coupling?
c) Why do we need design patterns? Explain.

(6+6+6)

6.

a) What is change control process? Explain.
b) What are the reasons to opt for reverse engineering?
c) Describe the building blocks of Use Case Diagram?

(6+6+6)

7.

a) What are software metrics and software measurement?

b) Explain system design. What are physical and logical designs?

c) Do you think that testing object-oriented programs is easier than testing procedural programs? Explain your answer with special mention as to how the object-orientation features of inheritance, encapsulation, polymorphism and dynamic binding influence effective test case design.

(6+6+6)